Internet Programming in Python - Week 6

Small web frameworks

Re-inventing the wheel is great if your goal is to learn more about wheels.
-- James Tauber

Brian Dorsey
brian@dorseys.org

ANNOUNCEMENTS

Portland Python sprint Saturday, February 26th

http://goo.gl/Ln68F

http://
datacampseattle.eventbrite.com/

SeaPIG tomorrow!

Cython

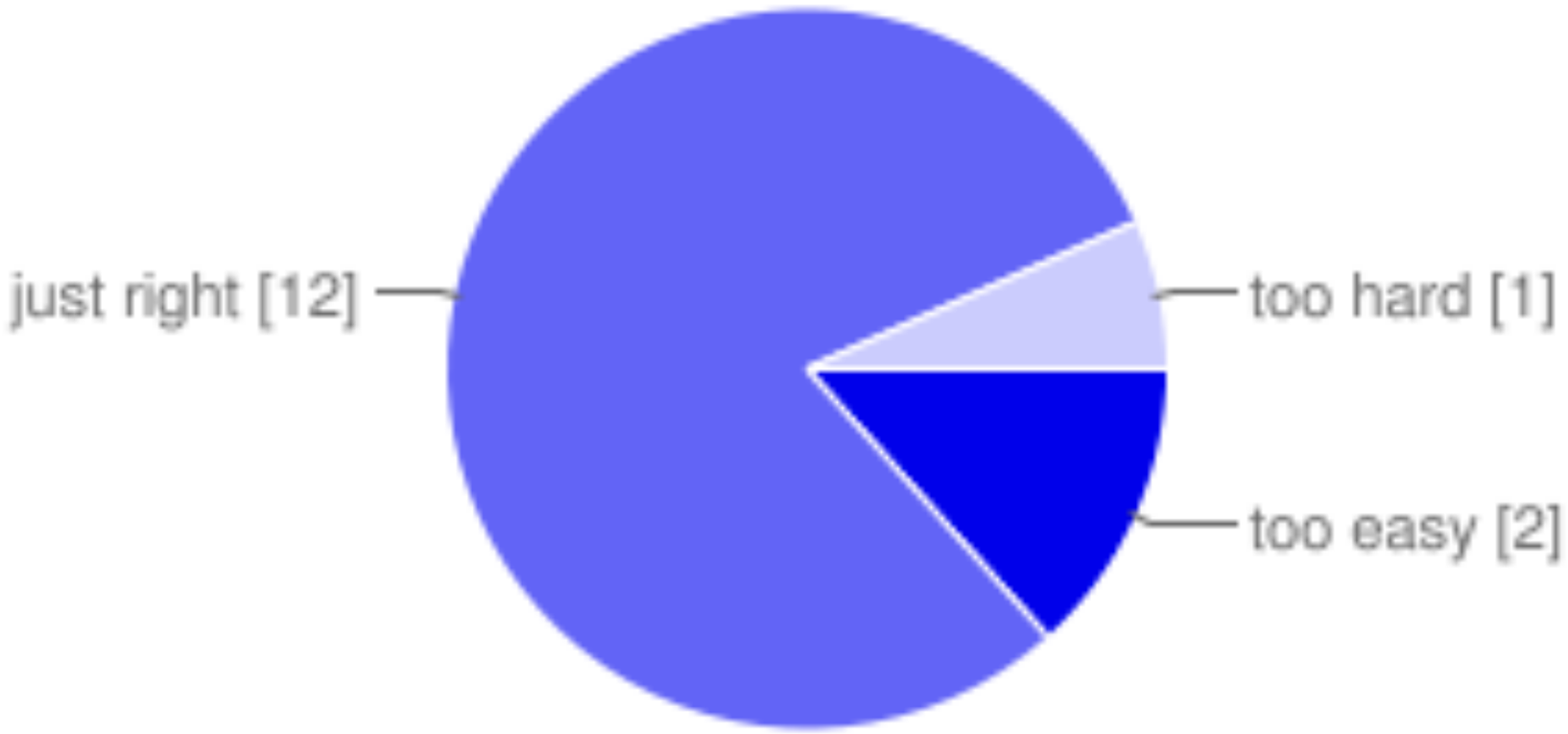
http://us.pycon.org/2011/home/

DjangoCon
probably Portland in
September

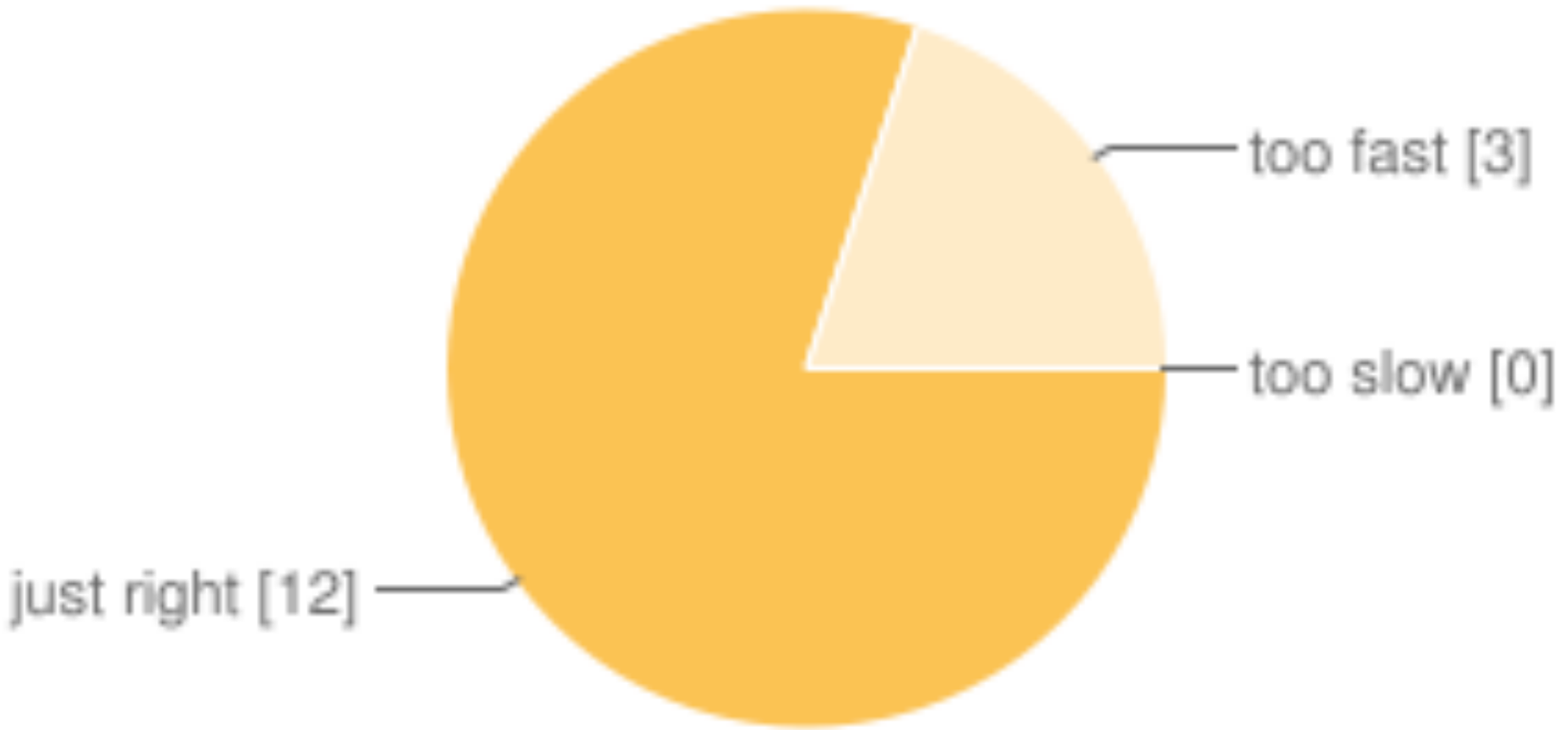
A moment to reflect

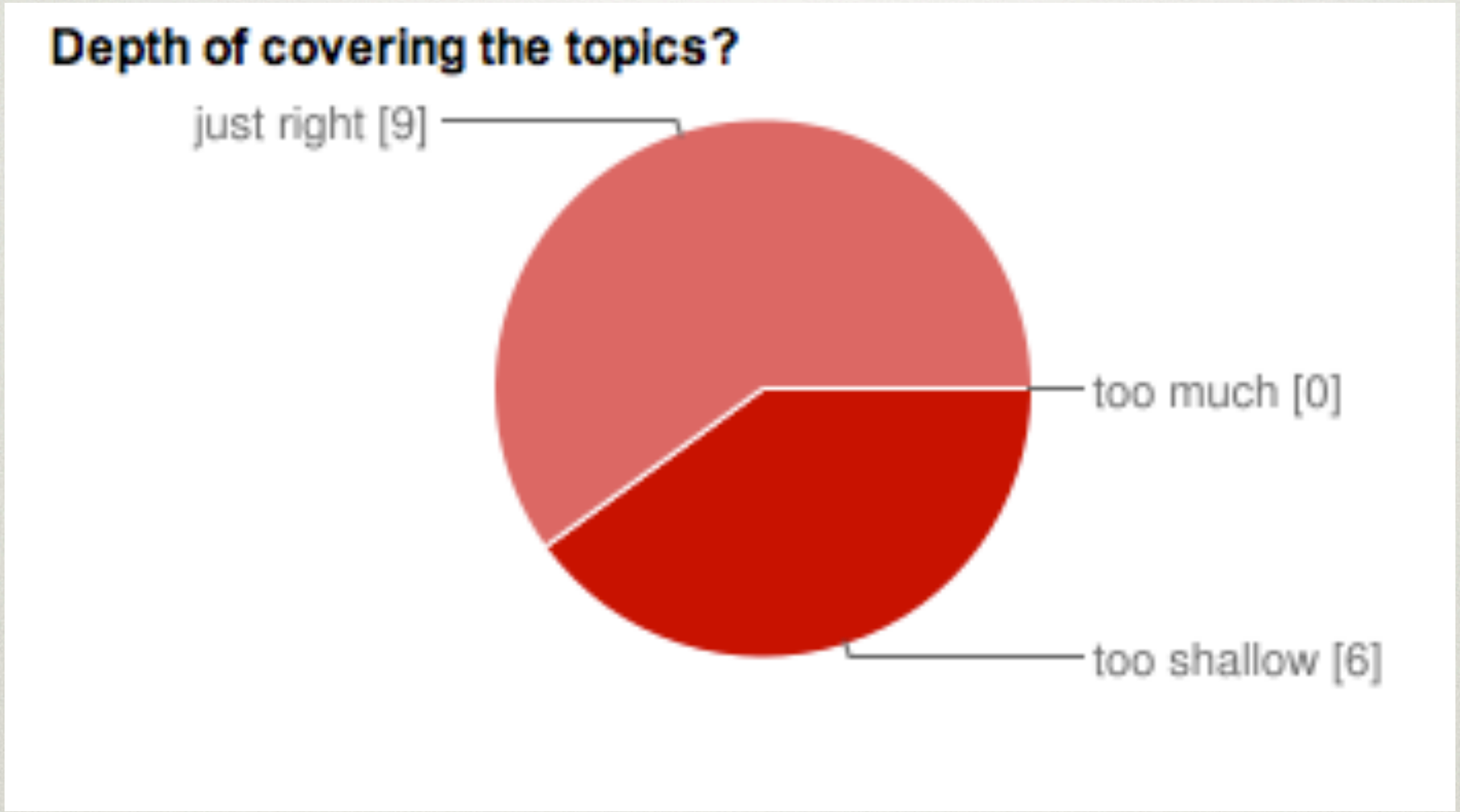
survey

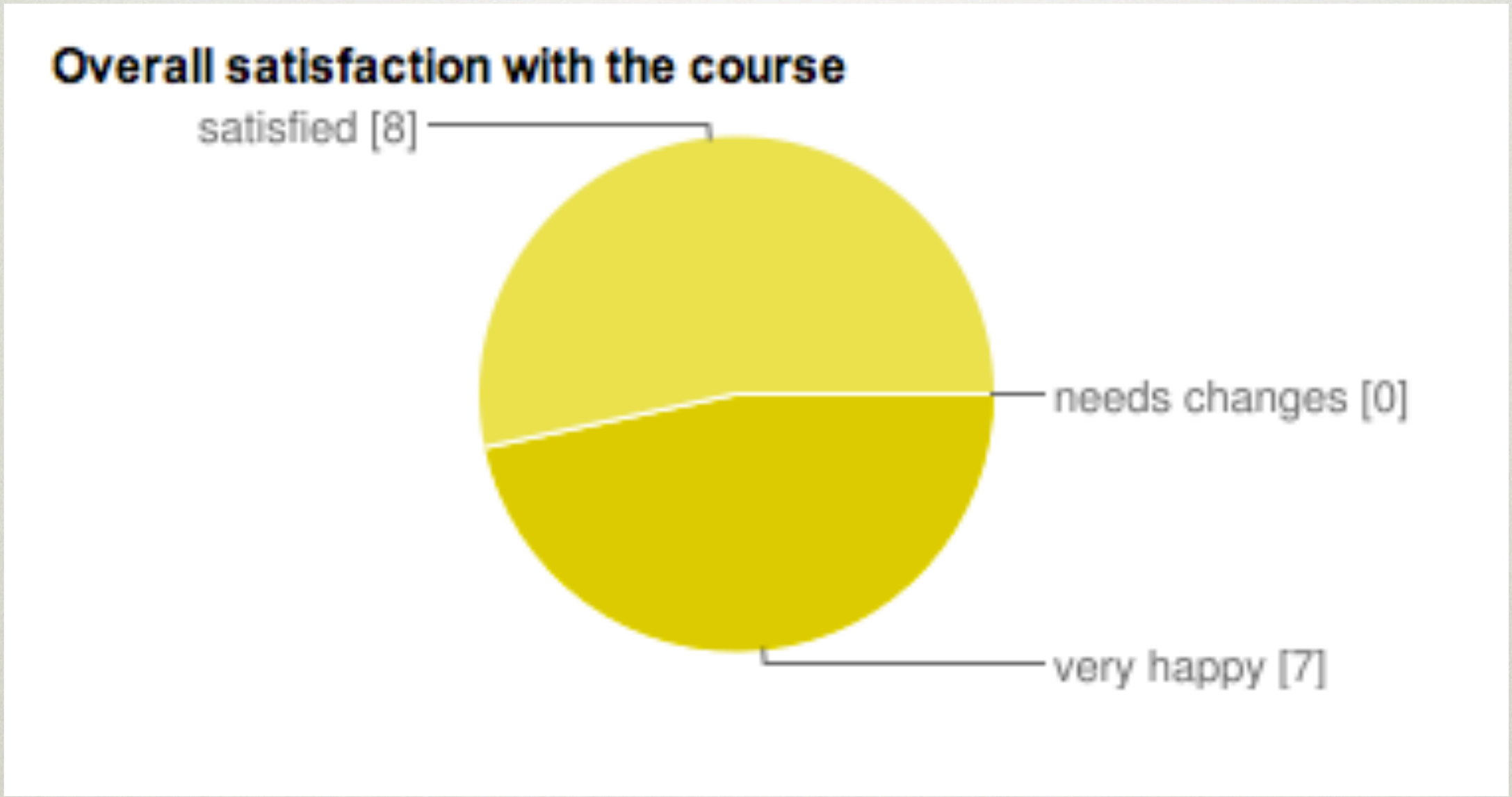
Difficulty of the material?

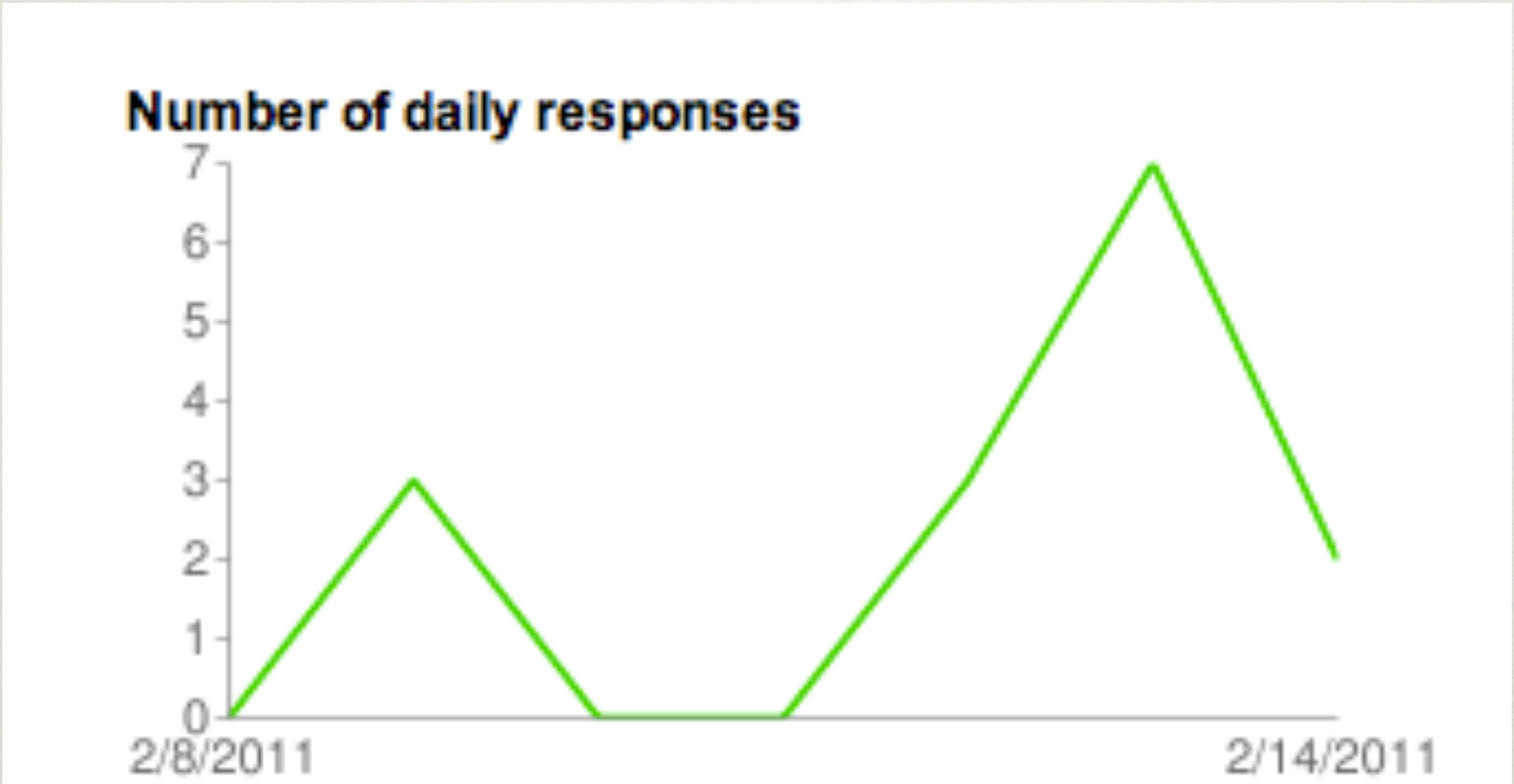


Pace of presenting the material?




Depth of covering the topics?
just right [9]
too much [0]
too shallow [6]


Overall satisfaction with the course
satisfied [8]
needs changes [0]
very happy [7]



(we mostly work on weekends)

comments

lots of nice comments
thanks!

(but I won't make you read
them all here) :)

assignments

secondary objectives?

=

stretch

too shallow

homework is hard

forced to figure a lot of things out

yep

week 9 topics

testing?

databases?

build whole app in class?

Questions and Review (20)

some thoughts from the assignments

turn in here: http://bit.ly/uwipip_week5

Assignment

- write a multi page website, using **`bookdb.py`** (see **`uwpython_web`** git repo)
- index page is a list of books, link to a detail page per book
- each detail page displays info about one book
- use any Python tech you'd like
- it's probably easier if you do the week 6 readings, *then* this assignment
- fill out feedback form: **`http://bit.ly/uwipip_midquarter`**

Library

Welcome to the amazing python book library!

Click on a link below for more information about the book

1. Python Cookbook, Second Edition
2. The Pragmatic Programmer: From Journeyman to Master
3. Python for Software Design: How to Think Like a Computer Scientist
4. Foundations of Python Network Programming
5. CherryPy Essentials: Rapid Python Web Application Development

how’d it go?

what did you use to implement it?

URLs

(pretty much anything works fine)

```
:8080/4
:8000/id4
:8080/id4/
:8081/id4/
:8080/book/id4
:8001/book/id4
:8082/title/id4
:8000/detail/id4
:8000/details/id4
:8080/book?id=id4
:8080/detail?id=id4
:8080/selected_book?id=id4
/book4/
/books/id4
/books/details?id=id4
/books/book?id=id4
/seriously-who-puts-something-like-this-on-a-dev-server/
thats-a-bad-hack/books/id4/
```

```
/4
/book/id4
/book/id4
/book4/
/book?id=id4
/books/book?id=id4
/books/details?id=id4
/books/id4
/books/id4/
/detail/id4
/detail?id=id4
/details/id4
/id4
/id4/
/id4/
/selected_book?id=id4
/title/id4
```

some opinions

/4 *
/book/id4
/book/id4/   <-- my favorite
/book4/ *
/book?id=id4
/books/book?id=id4
/books/details?id=id4
/books/id4
/books/id4/
/detail/id4
/detail?id=id4
/details/id4
/id4
/id4/
/selected_book?id=id4
/title/id4

```
tools.trailing_slash.extra = False
```

be consistent within a site

pages

{'publisher': 'OReilly Media', 'author': 'Alex Martelli, Anna Ravenscroft, David Ascher', 'isbn': '978-0-596-00797-3', 'title': 'Python Cookbook, Second Edition'}

{'publisher': 'OReilly Media', 'author': 'Alex Martelli, Anna Ravenscroft, David Ascher', 'isbn': '978-0-596-00797-3', 'title': 'Python Cookbook, Second Edition'}

publisher: OReilly Media
author: Alex Martelli, Anna Ravenscroft, David Ascher
isbn: 978-0-596-00797-3
title: Python Cookbook, Second Edition
home

Book Details

Title: Python Cookbook, Second Edition

ISBN: 978-0-596-00797-3
Publisher: OReilly Media
Author: Alex Martelli, Anna Ravenscroft, David Ascher

Return to List

publisher: OReilly Media
author: Alex Martelli, Anna Ravenscroft, David Ascher
isbn: 978-0-596-00797-3
title: Python Cookbook, Second Edition
Home

Title: Python Cookbook, Second Edition
ISBN: 978-0-596-00797-3
publisher: OReilly Media
author Alex Martelli, Anna Ravenscroft, David Ascher

Build time: 0.021s, Page size: 0.52KB

Python Cookbook, Second Edition by Alex Martelli, Anna Ravenscroft, David Ascher

ISBN: 978-0-596-00797-3 Publisher: OReilly Media

- Title : Python Cookbook, Second Edition
- ISBN : 978-0-596-00797-3
- Publisher : OReilly Media
- Author : Alex Martelli, Anna Ravenscroft, David Ascher

[Return]

Title: Python Cookbook, Second Edition
Publisher: OReilly Media
author: Alex Martelli, Anna Ravenscroft, David Ascher
ISBN: 978-0-596-00797-3
back

Title: Python Cookbook, Second Edition
Isbn: 978-0-596-00797-3
Publisher: OReilly Media
Author Alex Martelli, Anna Ravenscroft, David Ascher (back)

Python Cookbook, Second Edition

Alex Martelli, Anna Ravenscroft, David Ascher

OReilly Media

978-0-596-00797-3

Books
TitlePython Cookbook, Second Edition
Publisher
OReilly Media
ISBN
978-0-596-00797-3
Author
Alex Martelli, Anna Ravenscroft, David Ascher

"Title : CherryPy Essentials: Rapid Python Web Application Development "

"ISBN : 978-1904811848 "

"Publisher : Packt Publishing (March 31, 2007) "

"Author : Sylvain Hellegouarch "

Book Info

Here is the information you requested!

Title	Python Cookbook, Second Edition
Author	Alex Martelli, Anna Ravenscroft, David Ascher
Publisher	OReilly Media
ISBN	978-0-596-00797-3

Book Database

Python Cookbook, Second Edition

- ISBN: 978-0-596-00797-3
- Publisher: OReilly Media
- Author: Alex Martelli, Anna Ravenscroft, David Ascher

Back

Book detail

Python Cookbook, Second Edition
by
Alex Martelli, Anna Ravenscroft, David Ascher
OReilly Media, 978-0-596-00797-3

Title: Python Cookbook, Second Edition

Author: Alex Martelli, Anna Ravenscroft, David Ascher

Publisher: OReilly Media

ISBN: 978-0-596-00797-3

Back to book list

Book Details

title:	Python Cookbook, Second Edition
author:	Alex Martelli, Anna Ravenscroft, David Ascher
publisher:	OReilly Media
isbn:	978-0-596-00797-3

back to the titles

ZORG, A BOOK!

Title	Python Cookbook, Second Edition
Author	Alex Martelli, Anna Ravenscroft, David Ascher
Publisher	OReilly Media
ISBN	978-0-596-00797-3

Back to book list

Lecture A
(25)

web frameworks

we're moving up the stack

full web server

(thirty minute web server)

CGI

WSGI

(and the many other options for long-running processes)

from here on, we can think of everything else sitting on top of WSGI

it may not *actually* be true, but they're all working at that level of abstraction

so... what are the options?

Django
Grok
Pylons
TurboGears
web2py
Zope
CubicWeb
Enamel
Gizmo(QP)
Glashammer
Karrigell
Nagare
notmm
Porcupine
QP
SkunkWeb
Spyce
Tipfy
Tornado
WebCore
web.py
Webware
Werkzeug
WHIFF
XPRESS
AppWsgi
Bobo
Bottle
CherryPy
circuits.web
Python Paste
PyWebLib
WebStack
Albatross
Aquarium
Divmod Nevow
Flask
JOTWeb2
Python Servlet Engine
Pyramid
Repoze.bfg
Quixote
Spiked
weblayer

(partial list)

http://wiki.python.org/moin/WebFrameworks

Python the language isn't
domain-specific.
Frameworks are, and so there's
naturally a larger set of approaches
and multiple ideas about how to
solve the problems.

-- Robert Brewer (CherryPy)

http://www.b-list.org/weblog/2007/feb/23/pycon-2007-web-frameworks-panel/

* **Django**
Grok
Pylons
TurboGears
* **web2py**
Zope
CubicWeb
Enamel
Gizmo(QP)
Glashammer
Karrigell
Nagare
notmm
Porcupine
QP
SkunkWeb
Spyce
Tipfy
Tornado
WebCore
* **web.py**
Webware
* **Werkzeug**
WHIFF
XPRESS
AppWsgi
Bobo
Bottle
* **CherryPy**
circuits.web
Python Paste
PyWebLib
WebStack
Albatross
Aquarium
Divmod Nevow
Flask
JOTWeb2
Python Servlet Engine
* **Pyramid**
Repoze.bfg
Quixote
Spiked
weblayer

probably hundreds (thousands?) more internal frameworks used at various companies

full-stack frameworks

small frameworks

tools

plumbing

plumbing

sockets

HTTP

CGI

WSGI

tools

cgitb

WSGI middleware

CherryPy tools

Django middleware

small web frameworks

CherryPy
web2py
web.py
Werkzeug
many, many more...

full-stack frameworks

Django
Pyramid

full-stack frameworks

we're here ---> today

small frameworks

tools

plumbing

CherryPy

CherryPy is trying to scratch a smaller itch. It deliberately tries to be small enough to keep in your head, but also to enable people to scratch their own itches. ... CherryPy tries to do one thing — HTTP — and do it well.

-- Robert Brewer (CherryPy)

http://www.b-list.org/weblog/2007/feb/23/pycon-2007-web-frameworks-panel/

HTTP <--> normal Python

```
import cherrypy

class HelloWorld:
    def index(self):
        return "Hello world!"
    index.exposed = True

cherrypy.quickstart(HelloWorld())
```

http://www.cherrypy.org/wiki/CherryPyTutorial

are we cool with classes?

(quick demo)

did the CherryPy dispatch make sense?

/trunk/branch/leaf

root.trunk.branch.leaf

```
def(a):
```

```
def(a=None):
```

```
def(*args):
```

```
def(**kwargs):
```

(quick demo?)

Lab A

(20)

`git clone git://github.com/briandorsey/uwpython_web.git`
`(or, if you already have a copy: git pull)`

Lab A

- run **`lab_a_cherrypy_dispatch.py`**
  (running locally will be easier than VM)
- visit the site, and follow links
- fix the app when it crashes, or link targets are missing

- too easy? - make the server accept any path with any parameters, return all as JSON

solution

Break

(10)

API Talks (20)

talks

Guest Speaker

(20)

Leo Parker Dirac

http://www.embracingchaos.com/

Break

(10)

RANDOM

http://www.python.org/doc/humor/

Lecture B (20)

dispatching

trees of objects

ex.

```
root = HelloWorld()          # /
root.onepage = Page()        # /onepage/
root.otherpage = Page()      # /otherpage/

# usually this is done in the class
root.onepage.page = Page()  # /onepage/page/


cherrypy.quickstart(HelloWorld())
```

```
class Root:
    def index(self):
        ...
    index.exposed = True
```

```
class Root:
    @cherrypy.expose
    def index(self):
        ...
```

alternate dispatchers

RoutesDispatcher

ex.

```
d = cherrypy.dispatch.RoutesDispatcher()
d.connect(name='hounslow', route='hounslow', controller=City('Hounslow'))
d.connect(name='surbiton', route='surbiton', controller=City('Surbiton'),
          action='index', conditions=dict(method=['GET']))
d.mapper.connect('surbiton', controller='surbiton',
                 action='update', conditions=dict(method=['POST']))

conf = {'/': {'request.dispatch': d}}
cherrypy.tree.mount(root=None, config=conf)
```

http://docs.cherrypy.org/dev/concepts/dispatching.html

routes are explicit

no need for `@cherrypy.expose`

MethodDispatcher

ex.

```
class Root:
    exposed = True

    def __init__(self, *things):
        self.things = list(things)

    def GET(self):
        return repr(self.things)

    def POST(self, thing):
        self.things.append(thing)

root = Root(1, 2, 3)

d = cherrypy.dispatch.MethodDispatcher()
conf = {'/': {'request.dispatch': d}}
cherrypy.tree.mount(root, "/", conf)
```

http://www.cherrypy.org/wiki/PageHandlers

cherrypy._____

```
cherrypy.config

cherrypy.request
cherrypy.request.headers

cherrypy.response
cherrypy.response.headers
```

```
cherrypy.request
cherrypy.request.params
cherrypy.request.headers
cherrypy.request.config

cherrypy.request.params['a']
cherrypy.request.headers['name']
```

http://www.cherrypy.org/wiki/RequestObject

```
cherrypy.response
cherrypy.response.headers
cherrypy.response.status
```

```
cherrypy.response.headers['name'] = 'value'
```

http://www.cherrypy.org/wiki/ResponseObject

errors

ex.

```
raise cherrypy.HTTPError(403)
```

```
raise cherrypy.HTTPError("403 Forbidden",
    "You are not allowed to access this resource.")
```

http://www.cherrypy.org/wiki/ErrorsAndExceptions

404

ex.

```
_cp_config = {'error_page.404':
        os.path.join(localDir, "static/index.html")}
```

http://www.cherrypy.org/wiki/ErrorsAndExceptions

redirects

ex.

```
raise cherrypy.HTTPRedirect("")

raise cherrypy.HTTPRedirect("/abs/path", 307)

raise cherrypy.HTTPRedirect(["path1",
                             "path2?a=1&b=2"], 301)
```

especially, see note about POST ---> http://www.cherrypy.org/wiki/ErrorsAndExceptions

config files
config at setup
config at class definition

http://docs.cherrypy.org/dev/concepts/config.html

global config - files

```
[global]
server.socket_host: "64.72.221.48"
server.socket_port: 80
```

http://docs.cherrypy.org/dev/concepts/config.html

global config - code

```
cherrypy.config.update({'server.socket_host': '64.72.221.48',
                        'server.socket_port': 80,
                       })
```

http://docs.cherrypy.org/dev/concepts/config.html

attach to class (or method)

```
#configfile.ini
[/path/to/page]
methods_with_bodies = ("POST", "PUT", "PROPPATCH")
```

```
#code.py
def page(self):
    return "Hello, world!"
page.exposed = True
# this code is actually broken without the config
# let's attach it directly
page._cp_config = {"request.methods_with_bodies":
                   ("POST", "PUT", "PROPPATCH")}
```

http://docs.cherrypy.org/dev/concepts/config.html

tools

tools.accept
tools.auth_basic
tools.auth_digest
tools.basic_auth
tools.caching
tools.decode
tools.digest_auth
tools.encode
tools.err_redirect
tools.etags
tools.expires
tools.flatten
tools.gzip
tools.ignore_headers
tools.log_headers
tools.log_tracebacks
tools.nsgmls
tools.proxy
tools.redirect
tools.referer
tools.response_headers
tools.session_auth
tools.staticdir
tools.staticfile
tools.sessions
tools.tidy
tools.trailing_slash
tools.xmlrpc

http://www.cherrypy.org/wiki/BuiltinTools

http://www.cherrypy.org/wiki/ZenOfCherryPy

Lab B

(20)

`git clone git://github.com/briandorsey/uwpython_web.git`
`(or, if you already have a copy: git pull)`

Lab B

- install Django 1.2.5 on your laptop (see syllabus for install instructions)
- do the assignment. :)

Don't leave before seeing this:

It worked!

Congratulations on your first Django-powered page.

Of course, you haven't actually done any work yet. Here's what to do next:

- If you plan to use a database, edit the `DATABASES` setting in `mysite/settings.py`.
- Start your first app by running `python mysite/manage.py startapp [appname]`.

You're seeing this message because you have `DEBUG = True` in your Django settings file and you haven't configured any URLs. Get to work!

turn in here: `http://is.gd/uwipip_week6`

Assignment

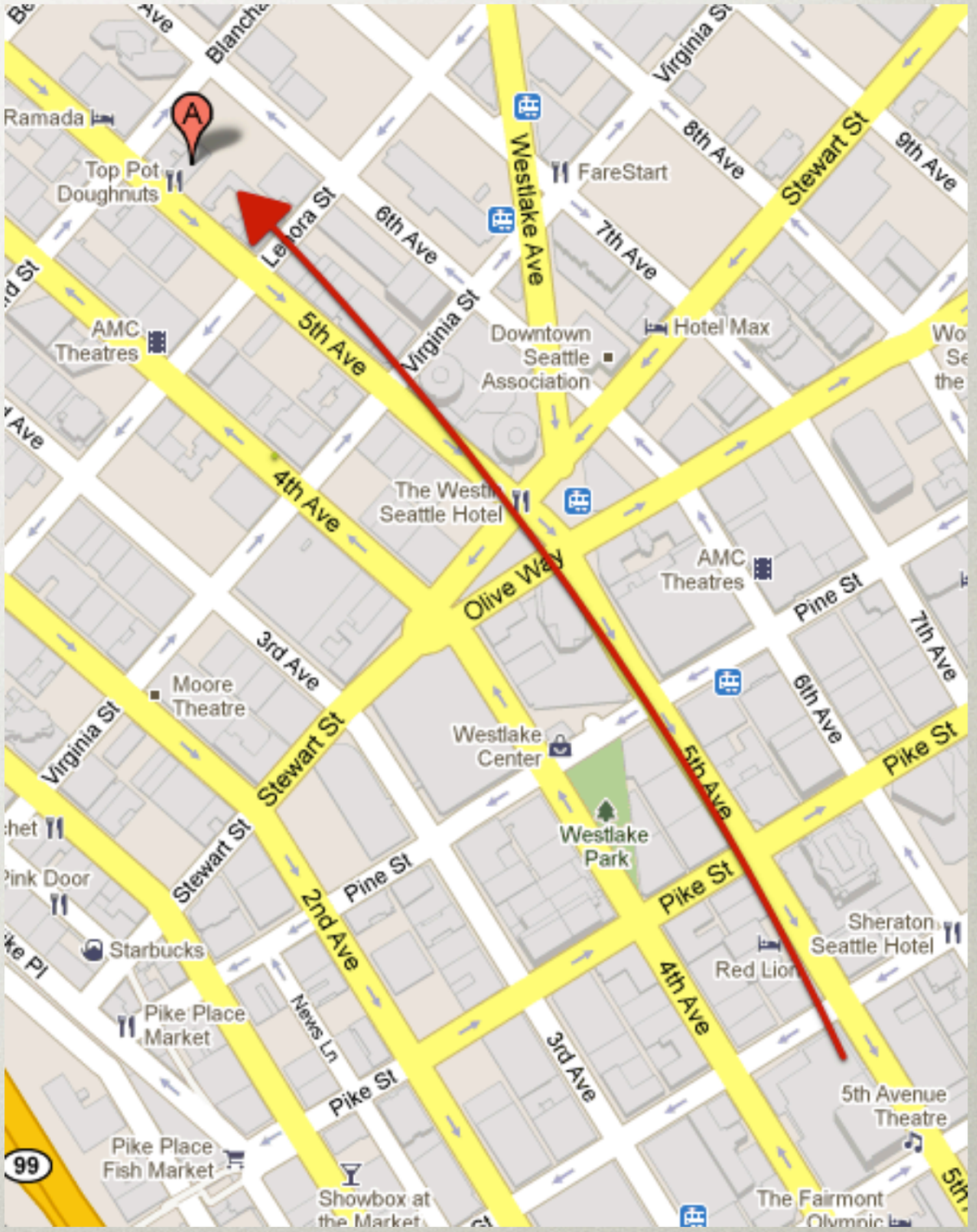
- week 6 assignment == week 7 reading
- do the Django tutorial (part 1, 2, 3): `http://docs.djangoproject.com/en/1.2/intro/tutorial01/`
- work through it locally, later push to VM
  ok to use development server:
  `python manage.py runserver 0.0.0.0:8000`
- stretch goal: also add and document a REST/JSON API for poll results

WRAPUP

Info

Office hours:
Sunday 2-5pm

Top Pot Donuts
2124 5th Ave
Seattle, WA 98121
206-728-1966



turn in here: `http://is.gd/uwipip_week6`

Assignment

- week 6 assignment == week 7 reading
- do the Django tutorial (part 1, 2, 3): `http://docs.djangoproject.com/en/1.2/intro/tutorial01/`
- work through it locally, later push to VM
  ok to use development server:
  `python manage.py runserver 0.0.0.0:8000`
- stretch goal: also add and document a REST/JSON API for poll results